DISPERSION MODULATION USING ALLPASS FILTERS

Jari Kleimola

Helsinki Institute for Information Technology (HIIT),
Helsinki University of Technology, TKK
Espoo, Finland
jari.kleimola@hiit.fi

ABSTRACT

Dispersion is a physical phenomenon that makes sound waves more or less inharmonic. Most physical sound synthesis models consider dispersion as a constant property that does not change during the course of a musical event. However, these models would be more expressive without such a restriction. This paper describes a dispersion amount parameter for precise control over inharmonicity, and then experiments with control and audio rate modulation of that parameter. In this research we found that inharmonicity of a plucked string could be smoothly controlled in real-time, and that novel sonic material could be synthesized when the modulation rate was raised into audio range. Instability of the string model with certain parameter values was considered to be problematic.

1. INTRODUCTION

In contrast to traditional sound synthesis methods that have their focus on the produced sound, physical modeling techniques concentrate on source vibrations and resonators of the musical instruments themselves. This often results in more accurate reproductions of real acoustic instrument timbres, and in some cases, allows generation of more artificial sounds (e.g., by combining excitations and resonating bodies from different instruments). It is also possible to modulate parts of the models that in real-world terms would be considered constant.

This paper explores the effects of control and audio rate modulation of dispersion, which in real-world would be in the constant domain, and thus beyond interactive or algorithmic control. It is hoped that such generalizations would give physical modeling techniques more power to generate previously unheard material – something that abstract synthesis techniques manage to do quite well.

The content of this paper is as follows. Section 2 describes the concept and physical properties of dispersion, discusses its modeling in digital waveguides (DWGs), and provides linkage to related works. Section 3 reviews the modulation sources that are commonly used in audio synthesis environments. Section 4 describes the implementation and testing setup, while section 5 shows the results and provides discussion. Finally, section 6 concludes the paper.[1]

[1] Sound examples used in this work are available at [1].

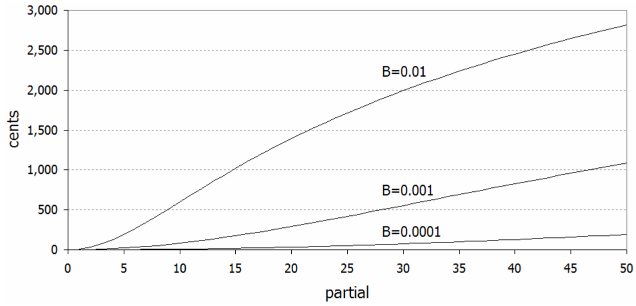

Figure 1: *The effect of dispersion to partial detuning, according to stretching equation (1).*

2. DISPERSION MODELING

Dispersion is a physical phenomenon, in which the propagation velocity of a wave is dependent on its frequency [2]. For sound waves traveling in a 1D waveguide (such as a plucked string), sinusoidal sound components (i.e., the partials) face different effective waveguide lengths. Therefore, partial frequencies are not exactly harmonic. If the amount of dispersion is low, the phenomenon can help in achieving more realistic generation of acoustical instrument timbres [3]. On the other hand, when the dispersion effect is exaggerated beyond real-world measures, it can produce metallic, bell-like, and other highly inharmonic synthetic timbres.

2.1. Dispersion in Strings

Dispersion in string instruments is caused by the stiffness of the string [4]. The result of the dispersion is that the partial frequencies f_n are stretched (cf. figure 1) according to equations

$$f_n \approx n f_0 \sqrt{1 + Bn^2} \tag{1}$$

$$B = \frac{\pi^3 E d^4}{64 L^2 T} \tag{2}$$

where n is the partial number, f_0 is the fundamental frequency of ideal string, B is the stretching factor, E is Young's modulus, d is the diameter, L is the length, and T is the tension of the string [5]. E depends on the material of the string. For nylon strings, where E is very small, the amount of stretching is not as prominent as for low register piano strings, where E is high.

2.2. Dispersion in Acoustic Tubes

Stretching of partials in strings is a result of *material* dispersion, i.e., it is dependent on the structural properties of the string. Acoustic tubes share identical material dispersive properties, because air stiffness results in similar inharmonicity of partials [6]. However, this holds only when the tube is excited impulsively: if the tube is driven by a continuous excitation signal (like jet air in a clarinet or a flute), material dispersion does not have an effect on inharmonicity. Rather, continuous excitation interacts with the resonating characteristics of the instrument body itself, locking the modes, and producing dominant resonances that may or may not be harmonic. Thus, for acoustic tubes with continuous excitation signal, inharmonicity is caused by the geometrics and other properties of the tube (i.e., it is the result of *waveguide* dispersion). For example in a clarinet, the bore and tone holes make upper partials more flat than lower ones [7].

2.3. Allpass Filters

Material dispersion phenomenon of strings can be modeled by inserting an allpass filter into the feedback path of a DWG, as first described by Jaffe and Smith, 1983 [8]. An allpass filter is characterized by a flat magnitude response, but with an arbitrary phase response, i.e., frequency dependent delay [9].

The dispersion modeling allpass filter can be implemented either as a single high-order filter, or by a cascade of several low-order stages. Various design strategies have been introduced to match the desired dispersion qualities, including a computationally effective tunable solution of Rauhala and Välimäki [10,11].

3. MODULATION SOURCES

Control rate modulation of synthesis parameters can be done either by utilizing internal low frequency signal generators, or by responding to external performer gestures. A low frequency oscillator (LFO) operates usually within a frequency range of 0.01..20 Hz, often has a selectable waveshape, and sometimes, a delay parameter to postpone the onset of the modulation. An envelope generator (EG) produces exponential or piecewise linear functions of time, comprising multiple segments, each separately controllable by duration and amplitude parameters. Performer generated modulation events are usually results of continuous gestures that have relatively short timespans. Modulation wheels, pitch benders, ribbon or breath controllers, and keyboard aftertouch sensors are the most common input devices to produce such event streams.

When the modulating signal is raised so that it reaches audio frequencies, it is possible to synthesize complex timbres. For example, when a subtle control rate vibrato signal is used to modulate a carrier oscillator at audio rate (frequency modulation or FM), even two simple sinusoidal oscillators can produce a wide range of both harmonic and inharmonic timbres [12]. Another well-known example is ring modulation, where audio rate modulation source is used to alter the amplitude of a carrier oscillator, resulting mostly inharmonic metallic timbres [13]. In both cases, the key parameters include modulator and carrier waveform selectors, and the frequency ratios between the interacting oscillators.

The amount of modulation parameter is shared by both control and audio rate modulation setups. When designing a modulation scheme one needs to further consider also how the modulation source parameter space is transformed into the target parameter space, and whether the target allows bipolar or unipolar modulation.

4. ALGORITHMS AND IMPLEMENTATION

Pure Data (PD) is a modular block-based synthesis environment, which enables end user programming and real-time processing of audio synthesis algorithms [14]. These algorithms are created graphically by interconnecting elementary synthesis blocks into acyclic graph structures, thereby describing the flow of control and audio rate signals.

Several reusable sound synthesis components are included in the PD distribution. The system can also be extended using external modules (DLLs in Windows environment) that can be developed or downloaded from the Internet, and installed into the component palette of PD. In order to achieve this kind of extensibility, PD defines an application programming interface in C, which allows development of arbitrarily complex synthesis elements.

This work utilizes both graphical patching and external programming. The graphical patch contains components and connections for synthesis parameter setting, performance input, control and audio rate function generators/modulators, and audio output section (cf. figure 2). The combined plucked string and dispersion model was implemented as a PD external (*displuck~*).

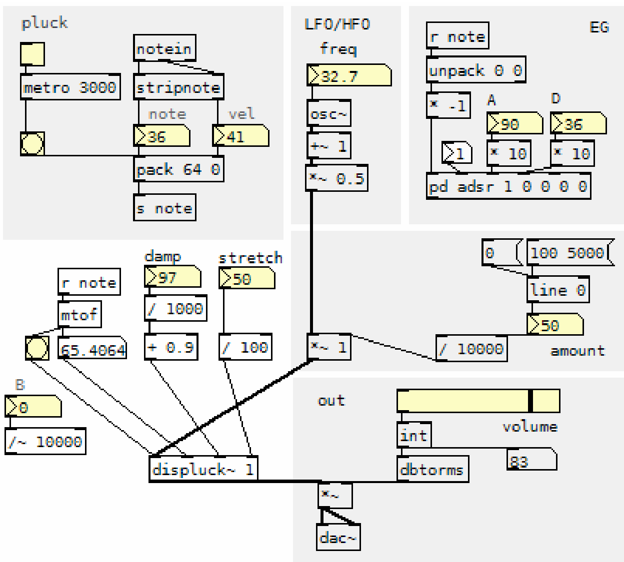

Figure 2: *PD implementation. The string and dispersion model is within the displuck~ external, also implemented in this work.*

4.1. Extended Karplus-Strong Algorithm

Jaffe and Smith extended the basic plucked string algorithm of Karplus and Strong [15] by providing the DSP formulation of the model, and by adding five filters to the structure in order to form a more realistic string model [8]. This Extended Karplus-Strong algorithm (EKS) is depicted in figure 3, where the added filters are denoted by H_c through H_g.

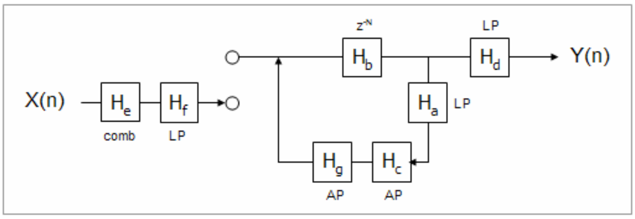

Figure 3: *Extended Karplus-Strong algorithm (adopted from [8]). LP denotes low pass and AP allpass filter. X(n) is the excitation signal, while Y(n) is the output.* H_g *is the dispersion allpass.*

The filters are as follows. H_a is the loss filter, and H_b is the delay line. H_c provides fractional tuning correction, H_d provides the dynamics, H_e models the effect of pick position, H_f models the difference between up and down picking, and finally, H_g is used to model the stiffness of the string, i.e., the dispersion.

The algorithm starts by feeding an excitation signal *X(n)* into the system. Excitation is a short white noise burst, which is routed through a comb and low pass filter cascade in order to simulate the effect of picking. Filtered excitation fills the delay line H_b, whose length defines coarse tuning of the string. After the delay has been filled, the excitation is disconnected from the system.

Next, the oldest sample is extracted from the delay line, and routed into the low pass filtered output. The oldest sample is also fed back into the system. In the feedback path, it is first averaged with a previously outputted sample in order to simulate losses and decay of harmonics. The resultant averaged sample is then further filtered by a fractional delay to fine-tune the pitch, and by a dispersion filter to simulate the stiffness of the string (filter design is not presented in [8]). Processed sample is finally stored into the front of the delay line, and the process is repeated for the next oldest sample retrieved from the back of the delay line. This continues repeatedly until all harmonics have faded into near silence.

4.2. Tunable Dispersion Filter Design Algorithm

Rauhala and Välimäki describe a tunable closed-form algorithm for dispersion filter design in [10]. Their solution was made for a cascade of four second order Thiran allpass filters for the low fundamental frequencies, and for a single second order Thiran allpass filter for the higher ones. In a later paper, they extend the formula to cascade an arbitrary number of first order Thirans to model the desired dispersion properties [11]. This work utilizes the later design, and uses a cascade of eight first-order Thirans for dispersion modeling.

The extended formula takes fundamental frequency f_0, dispersion coefficient B from equation 2, and the number of stages M as input parameters, and produces the allpass coefficient a_1 using equations 3...7, as shown below.

$$a_1 = \frac{1-D}{D+1} \tag{3}$$

$$D(I_{key}, B) = e^{(C_d(B) - I_{key} k_d(B))} \tag{4}$$

where D is the desired delay at DC, which can be calculated using

$$I_{key}(f_0) = \log_{\sqrt[12]{2}} \frac{f_0 \sqrt[12]{2}}{27.5} \tag{5}$$

$$C_d(B, M) = e^{(m_1 \ln M + m_2)\ln M + m_3 \ln M + m_4)} \tag{6}$$

$$k_d(B) = e^{(k_1 (\ln B)^2 + k_2 \ln B + k_3)} \tag{7}$$

$m_1..m_4$ and $k_1..k_3$ are predefined constants. The dispersion filter adds extra delay to the feedback loop, so its effect should be compensated so that the string stays in tune. The approximate delay line length becomes $L_1 = \text{floor}(f_s / f_1 - MD) - 1$, while the corresponding fractional tuning allpass delay will be $d_c = f_s / f_1 - MD - L_1$. In previous equations, f_s is the sampling rate and f_1 is the frequency of the first partial.

4.3. Plucked String Instrument

The plucked string instrument was implemented as an Extended Karplus-Strong algorithm, without filters H_d, H_e, and H_f. The PD patch of the instrument is shown in figure 2. There are six sections in the patch. *pluck* generates (or interfaces an external MIDI controller to trigger) a note, *LFO/HFO*, *EG* and *amount* sections implement the modulators, *out* interfaces the audio outputs, and the rest of the patch is devoted to the *model* and its control parameters.

4.4. String and Dispersion Model Implementation

displuck~ external was programmed in C language. It has one signal inlet (dispersion value B), three float inlets (fundamental frequency, and two parameters affecting the length of the generated waveform), and one signal outlet emitting the synthesized signal. *displuck~* takes also a creation argument that defines the number of cascaded first order Thiran allpass filters in the dispersion model.

Banging the leftmost inlet fills an internal variable length delay line with noise. The length of the delay line has been determined by a routine that is invoked when the frequency of the second inlet changes. The routine calculates also a fractional value that is used to adjust the coefficient of the fine-tuning allpass filter H_c.

PD calls the processing function of *displuck~* repeatedly, which generates a block of samples during each invocation. First, it calculates the dispersion allpass coefficient from supplied B value using the algorithm described in section 4.2[2].

The routine returns also the dispersion allpass chain phase delay at DC, which must be compensated so that the tone stays in tune. This compensation is done by adjusting the internal variable length delay line and the fine tuning allpass filter. It should be noted that because dispersion value B can be modulated, this process is repeated for each block of samples.

[2] the calculation routine is a direct C-language port of the matlab algorithm referenced in [11].

The processing function then enters a loop which generates one sample per iteration. For each sample, the oldest entry is first read from the delay line and stored into the output buffer. It is then averaged with the previously outputted sample (according to damp and stretch parameters), routed through the fine-tuning allpass, and processed by the dispersion cascade. The result is stored in the delay line, and persisted for the next iteration step.

4.5. Measurements

Special attention was given to the following properties during the testing period. In order to propose a practical control target parameter mapping, various static values were used for the dispersion amount *B*. An attempt to compensate the impact of dispersion allpass to the overall length of the string model (i.e. tuning) was performed as well. Static, control and audio rate modulation tests were performed.

A single tone having fundamental frequency of 65.4 Hz (corresponding to C2) was used in testing. A white noise generator was used as the excitation source. The tests were conducted using a PC with off-the-shelf soundcard, 1 GB of RAM, and 1.59 GHz CPU clock rate.

5. RESULTS AND DISCUSSION

5.1. Static Properties

The value of the dispersion amount *B* was kept static, but different values ranging from 0.00001 (no dispersion) to 0.01 (extensive stretch amount: 8.61 cents for the fundamental) were tested. Effects of values below 0.0001 were not clearly audible, and generally, all values less than 0.001 were close enough to the original timbre to be perceived as coming from a stringed instrument. However, between 0.001 and 0.01 the tone morphed from the original plucked string sound towards bell-like timbres quite rapidly.

Tuning compensation was considered to be quite difficult. *B* values below 0.001 affected tuning very little, so compensation was relatively successful, but for values between 0.001 and 0.005, the perceived pitch seemed to be rising with the *B* value. However, frequency analysis proved that the pitch of the fundamental did not change (cf. figure 4), see also [3] for related discussion. The fundamental frequency could not be easily recognized for bell-like timbres, i.e., for *B* values greater than 0.005. With very high *B* and fundamental frequency values, the amount of needed compensation exceeds the length of the principal delay line.

The CPU load could not be tested reliably, because it was impossible to disable operating system task switching while synthesizing. However, Windows Task Manager did not report any significant increase in CPU usage when the PD patch was running.

5.2. Modulation Tests

Control rate modulation was tested using an LFO, EG and a mouse-sensitive control to simulate performer generated input. LFO rates from 0.01 to 20 Hz were used in control rate testing. Simple ADSR envelope generator with various levels and durations was used as well. It was possible to smoothly modulate the level of inharmonicity by using any of these modulation

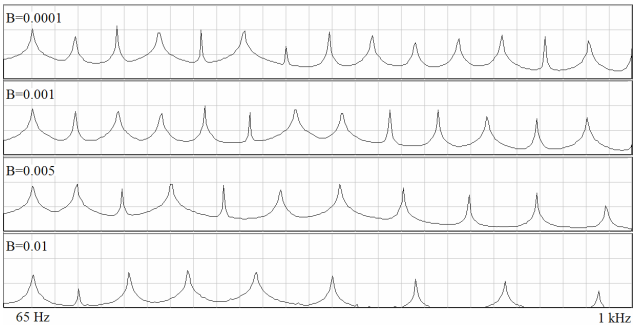

Figure 4: *The lower part of displuck~ spectrum (20..1000 Hz) with various values of dispersion amount B. Linear frequency scale, f_0 = 65.4 Hz.*

sources. Figure 5 depicts a spectrogram for an LFO-modulated timbre, where *B* sweeps between 0 and 0.01 at the rate of 1 Hz. White noise was used for string model excitation.

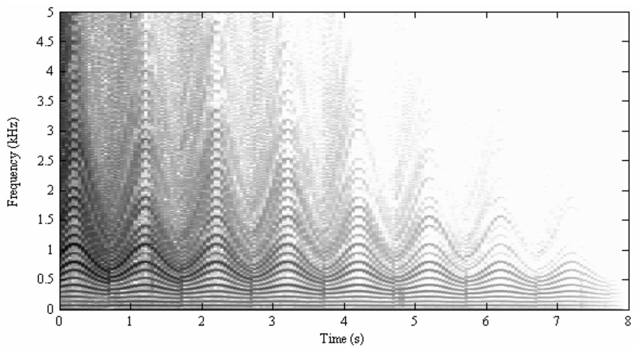

Figure 5: *Control rate modulation of dispersion amount. B is sweeped between 0 and 0.01, using a sinusoidal LFO with 1 Hz rate as the modulating source.*

Audio rate modulation was tested with sinusoidal modulator using rates from 20 Hz upwards, and modulating *B* values from 0.0001 to 0.01. The resulting signal was quite complex (cf. figure 6), and therefore only one first-order Thiran allpass was used as the dispersion filter, instead of a cascade of eight used in other examples. Tuning compensation was also ignored.

Certain frequency ratios between the fundamental frequency *c* of the string model and the modulating oscillator *m* produced thick and powerful timbral qualities. These *m/c* ratios were near 0.5, 1.0, 1.5, 2.0 and so on. Especially for ratios above 1.0, the amount of modulation had to be kept small because the system became easily unstable near these ratios. The perceived effect added a chorus- or phaser-like body to the unprocessed plucked string sound, with warm and evolving quality. Figure 6 shows a spectrogram for *m/c* ratio 0.98, where the string model was excited with white noise. Sinusoidal excitation produced similar results, but with less harmonic content.

The reason for this effect is still unclear. The dispersion filter coefficient updates did not seem to produce transients, although the state of the filter had discontinuities at update intervals. Different frequencies travelling along the string might also be affected in different ways by this DWG FM scheme. Ref. [16] discusses a similar self-regeneration effect that appears when the length of a delay line is modulated using a frequency near the fundamental frequency of the string. It was also noticed that some intermittent ratios gave spring reverb -like timbres, as discussed in [17].

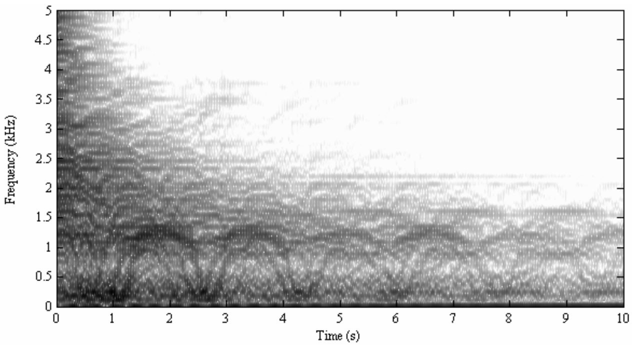

Figure 6: *Audio rate modulation of the dispersion allpass. White noise excitation, 65.4 Hz fundamental frequency, 64.4 Hz sinusoidal modulation, and maximum B deviation of 0.005.*

6. CONCLUSIONS AND FURTHER WORK

This paper experimented with control and audio rate modulation of dispersion in a DWG string model. It was possible to smoothly morph from nondispersive sounds into inharmonic timbres using LFO, EG and performer gestures. It seems that the range for the dispersion amount should be between 0.0001 and 0.01, and that the mapping should be logarithmic.

Audio rate modulation of dispersion amount parameter showed unexpected behaviour, and it was possible to generate timbres that had body, colour and movement that is not normally associated with simple plucked string models. The sound resembled chorused or phased polysynth/brass timbres, although it was produced entirely inside the string model. The audio rate modulator vs. fundamental frequency (i.e., the DWG length) ratio needed to be set to values 0.5, 1.0, 1.5, 2.0 and so on in order to activate the effect. At higher ratios and high B values the system became easily unstable, although that was not as evident with lower ratios.

Audio rate modulation and instability issues require further work. It would also be interesting to investigate whether modulation can be applied to other constants beside string stiffness, and whether it is possible to further generalize the DWG models for increased expressive power.

7. ACKNOWLEDGMENTS

The author would like to thank Prof. V. Välimäki, J. Pakarinen and H.-M. Lehtonen for valuable suggestions and feedback, A. Haghparast and J. Kestilä for commenting an early version of this paper, and Ann Morrison for proofreading.

This work has been co-funded by the European Union (EU) as part of the 6th Framework Research Programme with the project CALLAS (ref. 034800).

8. REFERENCES

[1] *Sound examples* http://www.hiit.fi/u/kleimola/dsp/disp/dispmod.html

[2] ATIS, *ATIS Telecom Glossary 2007*, http://www.atis.org/glossary/definition.aspx?id=7348, Accessed June 08, 2008.

[3] H. Järveläinen, V. Välimäki, M. Karjalainen, "Audibility of the timbral effects of inharmonicity in stringed instrument tones", in *Acoustic Research Letters Online*, Vol. 2 (2001), Issue 3, pp. 79-84.

[4] H. Fletcher, E.D. Blackham, and R. Stratton, "Quality of piano tones", *J. Acoust. Soc. Am.*, 34(1962)6, 749-761.

[5] N.H. Fletcher, "The nonlinear physics of musical instruments", *Rep. Prog. Phys,*. 62(1999) 723–764.

[6] J.O. Smith, and D. Rocchesso, "The Ideal Acoustic Tube", in *Aspects of Digital Waveguide Networks for Acoustic Modeling Applications*, online book, Available at http://ccrma.stanford.edu/~jos/wgj/Ideal_Acoustic_Tube.html, Accessed June 08, 2008.

[7] J. Cotler, "Notes on vibrato, along with the physical properties of the clarinet", *Klarinet*, Feb 1995, Available at http://www.woodwind.org/clarinet/Study/Vibrato.html, Accessed June 08, 2008.

[8] D.A. Jaffe, and J.O. Smith, "Extensions of the Karplus-Strong Plucked-String Algorithm", *Computer Music Journal*, Vol. 7, No. 2 (Summer, 1983), pp. 56-69.

[9] J.O. Smith, *Physical Audio Signal Processing*, Aug. 2007 Edition, online book, http://ccrma.stanford.edu/~jos/pasp/ Accessed June 08, 2008.

[10] J. Rauhala, and V. Välimäki, "Tunable dispersion filter design for piano synthesis," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 253–256, May 2006.

[11] J. Rauhala, and V. Välimäki, "Dispersion Modeling in Waveguide Piano Synthesis Using Tunable Allpass Filters" *Proc. of the 9th Int. Conference on Digital Audio Effects* (DAFx-06), Montreal, Canada, September 18-20, 2006, Available at www.dafx.ca/proceedings/papers/p_071.pdf, Accessed June 08, 2008.

[12] J. Chowning, "The Synthesis of Complex Audio Spectra by Means of Frequency Modulation", *J. Audio Eng. Soc.*, vol. 21, No. 7, pp. 526-534, September 1973.

[13] U. Zölzer, Ed., *DAFX – Digital Audio Effects*, J. Wiley & Sons, 2002.

[14] *Pure Data portal*, http://puredata.info/, Accessed June 08, 2008.

[15] K. Karplus, and A. Strong, "Digital Synthesis of Plucked-String and Drum Timbres", *Computer Music Journal*, Vol. 7, No. 2 (Summer, 1983), pp. 43-55.

[16] T. Stilson, "General Weirdness with the Karplus-Strong String", in *Proc. of the 1995 Intl. Computer Music Conf.*, Alberta, Canada, 1995.

[17] J.S. Abel, D.P. Berners, S. Costello, J.O. Smith, "Spring Reverb Emulation Using Dispersive Allpass Filters in a Waveguide Structure", *AES 121st Convention*, Preprint Number 6954, October 2006.